Name : **Arjun Paudel**

Student ID Number : **S2271954**

Subject : **Software Development For Data Science**

Module Code : **MMI226822**

Coursework 1

Submission Date : **6 December 2023**

Abstract

This reports is aim to provide the details exploratory data analysis of sales price of housing property from 2006 to 2010 in a small town in United State of America. This work was done on python juputer notebook with the help of different python libraries. Different observation were analyzed over an eighty-two different variables. First the dataset were read from the google cloud and and merge the two text files in a single dataset with the help of pandas library. Different data processing technique were executed like cleaning null values, deleting duplicate data and eradicating different outlier present on the dataset. Then different visualize technique were introduce like bar plot, line graph, scatter plot, histogram, heat map and so on. And these visualize was done on both univariate and bivariate analysis.

Introduction

Background

House price dataset is a dataset that describes the sales of individual residental properties in a small town of United State of America on an interval of 2006 to 2010. There are all together 2939 obeservation and 82 variable after combining the two text files that was given as a dataset.

This dataset is a combination of four type of data, Nominal, Ordinal, Discrete, and Continuous. Some of the variable that comes under the nominal data are PID (Parcel Identification Number), MS SubClass (Types of dewlling involced in sale), MS Zoing (Identification of general zoing classification of sales), Roof Style and so on. Variables that comes under the Ordinal data are HeatingQC (Heating quality and conditions), utilities (types of utilities avilable) and so on. Discrete data variables are Year Built (construction date of property), Full Bath (Number of full bathrom above grade), Fireplaces (Number of total fireplace in house) and so on. And the variable that comes under continuous data are

Garage Area (Total area of garage in square feet), Misc Val (Total value of miscellaneous feature in dollar), SalePrice (sales price of house in dollar) and so on.

Objectives

The main objectives of this analysis is to understand the dataset and identify if there are any patterns or trends on the data so that it will be more easy to make a decision while buying a house. And another main aim form this exploratory data analysis is to find any relationship between different variables and calculating correlation between these variables.

DATA CLEANING / WRANGLING / MUNGING / TRANSFORMATION

Import Necessary Libraries

In [155…
```
#Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pylab as plt
import seaborn as sns
```

These are the four powerfull libraries which are used to analyze and visualize of the data in python. Where pandas library, is used for data manipulation and analysis, numpy is specially used for arthimetic operations and working with array, matplotlin.pylab is mainly used for plotting interface and also provide the more control over plot and seaborn is a statistical data visualization library which is used for attractive and informatic graphs.

In [164…
```
#To show all the column in the results
pd.set_option('display.max_columns', 100)

#To show all the rows in the results
pd.set_option('display.max_row', 100)
```

This code help to show all the variable and observations that were present in the dataset.

Data Ingestion

This code is used to mount the google drive into google colab which helps on reading and writing the file on google colab directly from drive.

In [165…
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call driv
e.mount("/content/drive", force_remount=True).
```

Reading the first dataset that is Housing_1.txt files from the google drive by using pandas library and delimiter = '\t' is used to indicate that the reading data is in tap seperated. And at last Housing1df.head() is used to display few first rows from the loaded dataset.

In [166…
```
# Specify the file paths where the data is avilable
Housing1 = '/content/drive/MyDrive/SDFDS/Housing_1.txt'

# Load the dataset with housing_1 files data
Housing1df = pd.read_csv(Housing1, delimiter='\t')

# Display the first few rows of the dataset
Housing1df.head()
```

Out[166]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Land Contour	Utilities
0	1	526301100	20	RL	141.0	31770	Pave	NaN	IR1	Lvl	AllPub
1	2	526350040	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub
2	3	526351010	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub
3	4	526353030	20	RL	93.0	11160	Pave	NaN	Reg	Lvl	AllPub
4	5	527105010	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub

Loading the second dataset that is Housing_2.txt files from the google drive.

In [167…
```
# Specify the file paths where the second data is avilable
Housing2 = '/content/drive/MyDrive/SDFDS/Housing_2.txt'

# Load the dataset with housing_1 files data

Housing2df = pd.read_csv(Housing2, delimiter='\t')


# Display the first few rows of the dataset
Housing2df.head()
```

Out[167]:

	Order	PID	MS SubClass	SalePrice
0	1	526301100	20	215000
1	2	526350040	20	105000
2	3	526351010	20	172000
3	4	526353030	20	244000
4	5	527105010	60	189900

Now merging these two datset into the one single dataset with the help of common column in both dataset. Here PID, Order and MS SubClass columns are common on both dataset so the merge will be done with the help of these columns. Here another pieces of code

pd.set_option('display.max_columns', None) is used to display all the colums that are on our dataset.

In [169…
```
#Setout the common column
common = ['PID', 'Order', 'MS SubClass']

#Merging the two dataset  with the help of common cloumn
df = pd.merge(Housing1df, Housing2df, on = common)

#Display the all the columns of loaded dataset
pd.set_option('display.max_columns', None)

# Display the first few rows of the dataset
df.head()
```

Out[169]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Land Contour	Utilities
0	1	526301100	20	RL	141.0	31770	Pave	NaN	IR1	Lvl	AllPub
1	2	526350040	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub
2	3	526351010	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub
3	4	526353030	20	RL	93.0	11160	Pave	NaN	Reg	Lvl	AllPub
4	5	527105010	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub

Data Description

Before moving to the further, understand the datset by displaying few rows of data set. For this run the follwing code:-

In [170…
```
#displaying the first few rows of data
df.head()
```

Out[170]:

	Order	PID	MS SubClass	MS Zoning	Lot Frontage	Lot Area	Street	Alley	Lot Shape	Land Contour	Utilities
0	1	526301100	20	RL	141.0	31770	Pave	NaN	IR1	Lvl	AllPub
1	2	526350040	20	RH	80.0	11622	Pave	NaN	Reg	Lvl	AllPub
2	3	526351010	20	RL	81.0	14267	Pave	NaN	IR1	Lvl	AllPub
3	4	526353030	20	RL	93.0	11160	Pave	NaN	Reg	Lvl	AllPub
4	5	527105010	60	RL	74.0	13830	Pave	NaN	IR1	Lvl	AllPub

Finding out the number of observation and number of variable in the dataset.

In [171…
```
#shape of dataset that is number of rows and column
df.shape
```

Out[171]:
```
(2939, 82)
```

This shows that the dataset that has to be analyze has a total number of 2939 observation and 82 differernt variables.

Now displaying all the variables names.

In [172…
```
#displaying all the names of columns
df.columns
```

Out[172]:
```
Index(['Order', 'PID', 'MS SubClass', 'MS Zoning', 'Lot Frontage', 'Lot Area',
       'Street', 'Alley', 'Lot Shape', 'Land Contour', 'Utilities',
       'Lot Config', 'Land Slope', 'Neighborhood', 'Condition 1',
       'Condition 2', 'Bldg Type', 'House Style', 'Overall Qual',
       'Overall Cond', 'Year Built', 'Year Remod/Add', 'Roof Style',
       'Roof Matl', 'Exterior 1st', 'Exterior 2nd', 'Mas Vnr Type',
       'Mas Vnr Area', 'Exter Qual', 'Exter Cond', 'Foundation', 'Bsmt Qual',
       'Bsmt Cond', 'Bsmt Exposure', 'BsmtFin Type 1', 'BsmtFin SF 1',
       'BsmtFin Type 2', 'BsmtFin SF 2', 'Bsmt Unf SF', 'Total Bsmt SF',
       'Heating', 'Heating QC', 'Central Air', 'Electrical', '1st Flr SF',
       '2nd Flr SF', 'Low Qual Fin SF', 'Gr Liv Area', 'Bsmt Full Bath',
       'Bsmt Half Bath', 'Full Bath', 'Half Bath', 'Bedroom AbvGr',
       'Kitchen AbvGr', 'Kitchen Qual', 'TotRms AbvGrd', 'Functional',
       'Fireplaces', 'Fireplace Qu', 'Garage Type', 'Garage Yr Blt',
       'Garage Finish', 'Garage Cars', 'Garage Area', 'Garage Qual',
       'Garage Cond', 'Paved Drive', 'Wood Deck SF', 'Open Porch SF',
       'Enclosed Porch', '3Ssn Porch', 'Screen Porch', 'Pool Area', 'Pool QC',
       'Fence', 'Misc Feature', 'Misc Val', 'Mo Sold', 'Yr Sold', 'Sale Type',
       'Sale Condition', 'SalePrice'],
      dtype='object')
```

Now to see, what are the types of variable on these dataset, run this code.

In [173…
```
#displaying types of colums variable
df.dtypes
```

Out[173]:

```
Order              int64
PID                int64
MS SubClass        int64
MS Zoning         object
Lot Frontage     float64
Lot Area           int64
Street            object
Alley             object
Lot Shape         object
Land Contour      object
Utilities         object
Lot Config        object
Land Slope        object
Neighborhood      object
Condition 1       object
Condition 2       object
Bldg Type         object
House Style       object
Overall Qual       int64
Overall Cond       int64
Year Built         int64
Year Remod/Add     int64
Roof Style        object
Roof Matl         object
Exterior 1st      object
Exterior 2nd      object
Mas Vnr Type      object
Mas Vnr Area     float64
Exter Qual        object
Exter Cond        object
Foundation        object
Bsmt Qual         object
Bsmt Cond         object
Bsmt Exposure     object
BsmtFin Type 1    object
BsmtFin SF 1     float64
BsmtFin Type 2    object
BsmtFin SF 2     float64
Bsmt Unf SF      float64
Total Bsmt SF    float64
Heating           object
Heating QC        object
Central Air       object
Electrical        object
1st Flr SF         int64
2nd Flr SF         int64
Low Qual Fin SF    int64
Gr Liv Area        int64
Bsmt Full Bath   float64
Bsmt Half Bath   float64
Full Bath          int64
Half Bath          int64
Bedroom AbvGr      int64
Kitchen AbvGr      int64
Kitchen Qual      object
TotRms AbvGrd      int64
Functional        object
Fireplaces         int64
Fireplace Qu      object
Garage Type       object
Garage Yr Blt    float64
Garage Finish     object
Garage Cars      float64
Garage Area      float64
```

```
Garage Qual         object
Garage Cond         object
Paved Drive         object
Wood Deck SF         int64
Open Porch SF        int64
Enclosed Porch       int64
3Ssn Porch           int64
Screen Porch         int64
Pool Area            int64
Pool QC             object
Fence               object
Misc Feature        object
Misc Val             int64
Mo Sold              int64
Yr Sold              int64
Sale Type           object
Sale Condition      object
SalePrice            int64
dtype: object
```

To see the complete summary of the dataset before doing the data cleaning. This only show the values for integer and float types variable only.

In [174…
```
#displaying all the information from the dataframe that is means, total count, medi
df.describe()
```

Out[174]:

	Order	PID	MS SubClass	Lot Frontage	Lot Area	Overall Qual	Ove Co
count	2939.000000	2.939000e+03	2939.000000	2449.000000	2939.000000	2939.000000	2939.000(
mean	1461.462402	7.139155e+08	57.272882	69.264189	10147.527050	6.092548	5.561
std	847.802639	1.887018e+08	42.622774	23.334034	7868.008166	1.409733	1.110
min	1.000000	5.263011e+08	20.000000	21.000000	1300.000000	1.000000	1.000(
25%	726.500000	5.284770e+08	20.000000	59.000000	7447.500000	5.000000	5.000(
50%	1461.000000	5.354532e+08	50.000000	68.000000	9450.000000	6.000000	5.000(
75%	2195.500000	9.071801e+08	70.000000	80.000000	11523.000000	7.000000	6.000(
max	2930.000000	1.007100e+09	190.000000	313.000000	215245.000000	10.000000	9.000(

Data Cleaning / wrangling

In this chapter whole data cleaning process is defined where different task has been executed like handling the missing value, duplicates, or any outlier that has been present in the dataset.

Removing Null Values

In [175…
```
#checking the null values on whole datasets
df.isnull().any()
```

Out[175]:

Order	False
PID	False
MS SubClass	False
MS Zoning	False
Lot Frontage	True
Lot Area	False
Street	False
Alley	True
Lot Shape	False
Land Contour	False
Utilities	False
Lot Config	False
Land Slope	False
Neighborhood	False
Condition 1	False
Condition 2	False
Bldg Type	False
House Style	False
Overall Qual	False
Overall Cond	False
Year Built	False
Year Remod/Add	False
Roof Style	False
Roof Matl	False
Exterior 1st	False
Exterior 2nd	False
Mas Vnr Type	True
Mas Vnr Area	True
Exter Qual	False
Exter Cond	False
Foundation	False
Bsmt Qual	True
Bsmt Cond	True
Bsmt Exposure	True
BsmtFin Type 1	True
BsmtFin SF 1	True
BsmtFin Type 2	True
BsmtFin SF 2	True
Bsmt Unf SF	True
Total Bsmt SF	True
Heating	False
Heating QC	False
Central Air	False
Electrical	True
1st Flr SF	False
2nd Flr SF	False
Low Qual Fin SF	False
Gr Liv Area	False
Bsmt Full Bath	True
Bsmt Half Bath	True
Full Bath	False
Half Bath	False
Bedroom AbvGr	False
Kitchen AbvGr	False
Kitchen Qual	False
TotRms AbvGrd	False
Functional	False
Fireplaces	False
Fireplace Qu	True
Garage Type	True
Garage Yr Blt	True
Garage Finish	True
Garage Cars	True
Garage Area	True

```
Garage Qual         True
Garage Cond         True
Paved Drive        False
Wood Deck SF       False
Open Porch SF      False
Enclosed Porch     False
3Ssn Porch         False
Screen Porch       False
Pool Area          False
Pool QC             True
Fence               True
Misc Feature        True
Misc Val           False
Mo Sold            False
Yr Sold            False
Sale Type          False
Sale Condition     False
SalePrice          False
dtype: bool
```

Here **False** values mean no null values and **True** mean there are some null values on the variables. So to check the exact number of null values through out the dataset, run this code

In [176…

```
#checking exact number of null values in each variables
df.isnull().sum()
```

Out[176]:

Order	0
PID	0
MS SubClass	0
MS Zoning	0
Lot Frontage	490
Lot Area	0
Street	0
Alley	2741
Lot Shape	0
Land Contour	0
Utilities	0
Lot Config	0
Land Slope	0
Neighborhood	0
Condition 1	0
Condition 2	0
Bldg Type	0
House Style	0
Overall Qual	0
Overall Cond	0
Year Built	0
Year Remod/Add	0
Roof Style	0
Roof Matl	0
Exterior 1st	0
Exterior 2nd	0
Mas Vnr Type	23
Mas Vnr Area	23
Exter Qual	0
Exter Cond	0
Foundation	0
Bsmt Qual	80
Bsmt Cond	80
Bsmt Exposure	83
BsmtFin Type 1	80
BsmtFin SF 1	1
BsmtFin Type 2	81
BsmtFin SF 2	1
Bsmt Unf SF	1
Total Bsmt SF	1
Heating	0
Heating QC	0
Central Air	0
Electrical	1
1st Flr SF	0
2nd Flr SF	0
Low Qual Fin SF	0
Gr Liv Area	0
Bsmt Full Bath	2
Bsmt Half Bath	2
Full Bath	0
Half Bath	0
Bedroom AbvGr	0
Kitchen AbvGr	0
Kitchen Qual	0
TotRms AbvGrd	0
Functional	0
Fireplaces	0
Fireplace Qu	1425
Garage Type	157
Garage Yr Blt	159
Garage Finish	159
Garage Cars	1
Garage Area	1

```
Garage Qual         159
Garage Cond         159
Paved Drive           0
Wood Deck SF          0
Open Porch SF         0
Enclosed Porch        0
3Ssn Porch            0
Screen Porch          0
Pool Area             0
Pool QC            2926
Fence              2364
Misc Feature       2833
Misc Val              0
Mo Sold               0
Yr Sold               0
Sale Type             0
Sale Condition        0
SalePrice             0
dtype: int64
```

From the output it clearly shows that there are some variables which has the excessive number of null values. If we remove those values then this dataset will provide the false information while doing the analysis. So only those column has to be removed which has the less information. To do this just follow the next step.

In [177…

```
#selecting only most informative column only
df = df [['Order', 'PID', 'MS SubClass', 'MS Zoning',
          #'Lot Frontage',
          'Lot Area',
       'Street',
          #'Alley',
          #'Lot Shape',
          #'Land Contour',
          'Utilities',
       #'Lot Config', 'Land Slope',
          'Neighborhood',
          'Condition 1',
        'Condition 2',
          #'Bldg Type',
          'House Style', #'Overall Qual',
       'Overall Cond', 'Year Built', 'Year Remod/Add', 'Roof Style',
       'Roof Matl', 'Exterior 1st', 'Exterior 2nd', #'Mas Vnr Type',
       #'Mas Vnr Area', #'Exter Qual',
          'Exter Cond', 'Foundation',
          #'Bsmt Qual','Bsmt Cond', 'Bsmt Exposure',
          #'BsmtFin Type 1', 'BsmtFin SF 1',
       #'BsmtFin Type 2', 'BsmtFin SF 2', 'Bsmt Unf SF',
          'Total Bsmt SF',
       #'Heating',
          'Heating QC', 'Central Air', #'Electrical',
          '1st Flr SF',
       '2nd Flr SF',# 'Low Qual Fin SF',
          'Gr Liv Area', #'Bsmt Full Bath',
       #'Bsmt Half Bath',
          'Full Bath',
          #'Half Bath',
          'Bedroom AbvGr',
       'Kitchen AbvGr',
          #'Kitchen Qual',
          'TotRms AbvGrd',
          #'Functional',
       'Fireplaces',
```

```
        #'Fireplace Qu',
        #'Garage Type', 'Garage Yr Blt','Garage Finish',    'Garage Cars',
        'Garage Area',
        #'Garage Qual',       'Garage Cond', 'Paved Drive',
        'Wood Deck SF', 'Open Porch SF',
      #'Enclosed Porch', '3Ssn Porch', 'Screen Porch', 'Pool Area',
        #'Pool QC',    'Fence', 'Misc Feature',
        'Misc Val', 'Mo Sold', 'Yr Sold', 'Sale Type',
      'Sale Condition', 'SalePrice']].copy()
```

Here only those variables are selected which have the most important information. Only 40 variables has been selected and save it into the dataframe

In [178…
```
#seeing the total number of observation and variables.
df.shape
```

Out[178]:
```
(2939, 40)
```

Now checking is there is any null values on the new dataframe **df**.

In [179…
```
df.isnull().sum()
```

Out[179]:
```
Order             0
PID               0
MS SubClass       0
MS Zoning         0
Lot Area          0
Street            0
Utilities         0
Neighborhood      0
Condition 1       0
Condition 2       0
House Style       0
Overall Cond      0
Year Built        0
Year Remod/Add    0
Roof Style        0
Roof Matl         0
Exterior 1st      0
Exterior 2nd      0
Exter Cond        0
Foundation        0
Total Bsmt SF     1
Heating QC        0
Central Air       0
1st Flr SF        0
2nd Flr SF        0
Gr Liv Area       0
Full Bath         0
Bedroom AbvGr     0
Kitchen AbvGr     0
TotRms AbvGrd     0
Fireplaces        0
Garage Area       1
Wood Deck SF      0
Open Porch SF     0
Misc Val          0
Mo Sold           0
Yr Sold           0
Sale Type         0
Sale Condition    0
SalePrice         0
dtype: int64
```

In [180…
```
#showing the data from the missing values row
nullvalue = (df[df.isnull().any(axis =1)])
print(len(nullvalue))
print(df[df.isnull().any(axis =1)])
```

```
2
      Order        PID  MS SubClass MS Zoning  Lot Area Street Utilities  \
1350   1342  903230120           20        RM      5940   Pave    AllPub
2245   2237  910201180           70        RM      9060   Pave    AllPub

     Neighborhood Condition 1 Condition 2 House Style  Overall Cond  \
1350      BrkSide       Feedr        Norm      1Story             7
2245       IDOTRR        Norm        Norm      2Story             6

      Year Built  Year Remod/Add Roof Style Roof Matl Exterior 1st  \
1350        1946            1950      Gable   CompShg      MetalSd
2245        1923            1999      Gable   CompShg      Wd Sdng

     Exterior 2nd Exter Cond Foundation  Total Bsmt SF Heating QC Central Air  \
1350       CBlock         TA      PConc            NaN         TA           Y
2245      Plywood         TA     BrkTil          859.0         Ex           Y

      1st Flr SF  2nd Flr SF  Gr Liv Area  Full Bath  Bedroom AbvGr  \
1350         896           0          896          1              2
2245         942         886         1828          2              3

      Kitchen AbvGr  TotRms AbvGrd  Fireplaces  Garage Area  Wood Deck SF  \
1350              1              4           0        280.0             0
2245              1              6           0          NaN           174

      Open Porch SF  Misc Val  Mo Sold  Yr Sold Sale Type Sale Condition  \
1350              0         0        4     2008     ConLD        Abnorml
2245              0         0        3     2007       WD          Alloca

      SalePrice
1350      79000
2245     150909
```

Here two null values was show. So to remove these null values the following step was done

In [181…
```
#droping the na values
df = df.dropna(axis = 0)
```

All the null values that were present into the dataframe has been remove. SO to check this again to confirm the presence of null values.

In [182…
```
#checking again whether null values get eliminated or not
df.isnull().any()
```

Out[182]:
```
Order             False
PID               False
MS SubClass       False
MS Zoning         False
Lot Area          False
Street            False
Utilities         False
Neighborhood      False
Condition 1       False
Condition 2       False
House Style       False
Overall Cond      False
Year Built        False
Year Remod/Add    False
Roof Style        False
Roof Matl         False
Exterior 1st      False
Exterior 2nd      False
Exter Cond        False
Foundation        False
Total Bsmt SF     False
Heating QC        False
Central Air       False
1st Flr SF        False
2nd Flr SF        False
Gr Liv Area       False
Full Bath         False
Bedroom AbvGr     False
Kitchen AbvGr     False
TotRms AbvGrd     False
Fireplaces        False
Garage Area       False
Wood Deck SF      False
Open Porch SF     False
Misc Val          False
Mo Sold           False
Yr Sold           False
Sale Type         False
Sale Condition    False
SalePrice         False
dtype: bool
```

It is showing the false that means all the null values has been removed from the dataframe.

Removing Data Duplicate Values

First checking whether there are any duplicate data in the dataframe.

In [183…
```
#checking the duplicate data and showing the result by rows who has the duplicate d

df.loc[df.duplicated()]
```

Out[183]:

	Order	PID	MS SubClass	MS Zoning	Lot Area	Street	Utilities	Neighborhood	Condition 1	Con
146	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	
147	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	
148	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	
150	147	535175180	20	RL	10725	Pave	AllPub	NAmes	Norm	
151	147	535175180	20	RL	10725	Pave	AllPub	NAmes	Norm	
152	147	535175180	20	RL	10725	Pave	AllPub	NAmes	Norm	
154	148	535179020	20	RL	10032	Pave	AllPub	NAmes	Norm	
155	148	535179020	20	RL	10032	Pave	AllPub	NAmes	Norm	
156	148	535179020	20	RL	10032	Pave	AllPub	NAmes	Norm	

The result shows that there are few duplicate data in the dataframe. Also search query can be run to check the duplicate data that has the same PID. For this:-

In [184…

```
#checking of example of duplicate data through PID variable
df.query('PID == 535175070')
```

Out[184]:

	Order	PID	MS SubClass	MS Zoning	Lot Area	Street	Utilities	Neighborhood	Condition 1	Cond
145	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	N
146	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	N
147	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	N
148	146	535175070	20	RL	9300	Pave	AllPub	NAmes	Feedr	N

It shows all the **PID** who has the same value as **535175070**. If the duplicate values has larger number and we want the exact number of duplicate data following code will show the exact number of duplicate data.

In [185…

```
#checking total number of duplicate data according to the PID
df.duplicated(subset = ['PID']).sum()
```

Out[185]:

```
9
```

Now next step is to remove all these duplicate data from the dataframe.

In [186…

```
df = df.loc[~df.duplicated(subset = ['PID'])] \
     .reset_index(drop = True).copy()
```

This code remove the duplicate data with the help of **PID** variables from the dataframe. And also **reset_index(drop = True).copy()** helps to reset the index and copy function helps to create a new copy of the dataframe.</P>

To check whether the duplicate values has been removed or not just run the previous code which is used to check the duplicate values that is:-

In [187…
```
df.loc[df.duplicated()]
```

Out[187]:

Order	PID	MS SubClass	MS Zoning	Lot Area	Street	Utilities	Neighborhood	Condition 1	Condition 2	Hou St

This confirm that all the duplibate values has been removed from the data frame.

Cleaning Data Outlier

Another step of data processing is finding out any outliner present in the dataset and remove those outlier before doing any analysis.

First check if there is any outlier on the month variables. As there are only 12 months in a year but if the values of month is greater than 12 then that will be outlier.

In [188…
```
filtered_df = df[df['Mo Sold'] > 12]
filtered_df
```

Out[188]:

	Order	PID	MS SubClass	MS Zoning	Lot Area	Street	Utilities	Neighborhood	Condition 1	Con
541	542	531375130	20	RL	12450	Pave	AllPub	SawyerW	RRAe	
839	840	907131190	20	RL	14753	Pave	AllPub	CollgCr	PosN	

This result shows that there is two data which shows that the property was sold on the month of 60 and 120 which is never happend. SO this data is a outlier, and it has to be removed to get the better result from the analysis.

In [189…
```
#removing the outlier where month values is wrong

df = df[df['Mo Sold'] < 12]
print(len(df))
```

```
2823
```

This data has been removed

There is also a another outlier which also mentioned on the data dictionary, that the value of Gr Liv Area (living area above the groud in square feet) which has the higher than 4000. So inorder to remove this outlier follw the following step.

In [190…
```
filtered_grlivarea = df[df['Gr Liv Area'] >= 4000]
filtered_grlivarea
```

Out[190]:

	Order	PID	MS SubClass	MS Zoning	Lot Area	Street	Utilities	Neighborhood	Condition 1	Co
1497	1499	908154235	60	RL	63887	Pave	AllPub	Edwards	Feedr	
1759	1761	528320050	60	RL	15623	Pave	AllPub	NoRidge	Norm	
1766	1768	528351010	60	RL	21535	Pave	AllPub	NoRidge	Norm	
2179	2181	908154195	20	RL	39290	Pave	AllPub	Edwards	Norm	
2180	2182	908154205	60	RL	40094	Pave	AllPub	Edwards	PosN	

It shows that there are five data which has the execessive values of Gr Liv Area, it has to be removed before doing the analytics

In [191…
```
df = df[df['Gr Liv Area'] <= 4000]
print(len(df))
```

```
2818
```

These data also removed from the data frame. We have a clear dataframe without any missing values, outlier and duplicate data

DATA EXPLORATION AND ANALYSIS

Summary Statistics

Here is the summary statistics of the dataset.

In [192…
```
df.describe()
```

Out[192]:

	Order	PID	MS SubClass	Lot Area	Overall Cond	Year Built	Y Remod/A
count	2818.000000	2.818000e+03	2818.000000	2818.000000	2818.000000	2818.000000	2818.000
mean	1456.924769	7.147865e+08	57.219659	10108.709368	5.569553	1971.243080	1984.210
std	849.639186	1.887461e+08	42.550649	7803.704313	1.110646	30.233551	20.895
min	1.000000	5.263011e+08	20.000000	1300.000000	1.000000	1875.000000	1950.000
25%	723.250000	5.284770e+08	20.000000	7442.250000	5.000000	1953.000000	1965.000
50%	1455.500000	5.354541e+08	50.000000	9450.000000	5.000000	1973.000000	1993.000
75%	2190.750000	9.071811e+08	70.000000	11514.250000	6.000000	2001.000000	2004.000
max	2930.000000	1.007100e+09	190.000000	215245.000000	9.000000	2010.000000	2010.000

This is the actual summary of the dataset, where each variables has its total count, mean, standard deviation, minimum values, mean, median, and mode.

Data Visualisation

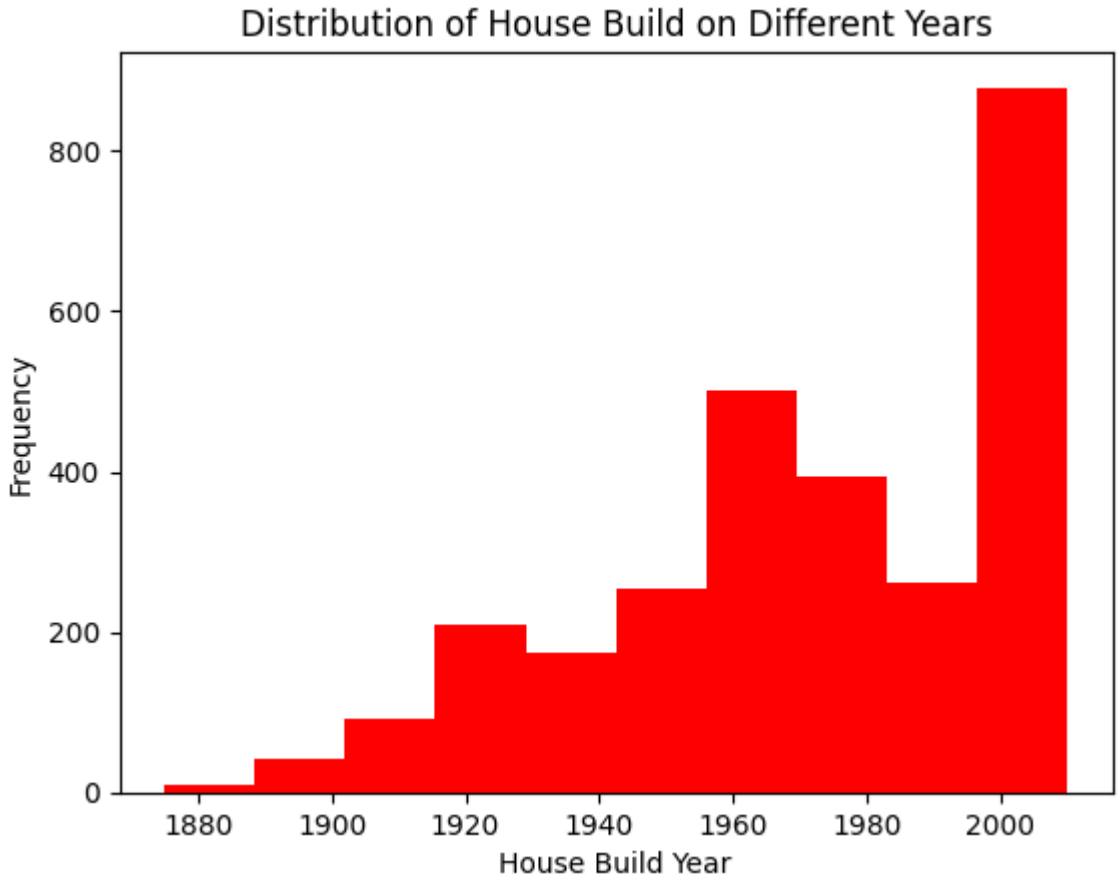
In this part data is analyze and visualize by using different visualization techniques with the help of different libraries. Also the visualization is based on two analysis that univariate (only one variable) and bivariate (using two or more than two variables) analysis.

Univariate Analysis

Question 1: Visualize the data on the basis of house build on the different year? (Visualize histogram graph by matplotlib library)

In [193…
```
ax = df['Year Built'].plot.hist( color = 'red')
ax.set_title('Distribution of House Build on Different Years')
ax.set_xlabel('House Build Year')
```

Out[193]:
```
Text(0.5, 0, 'House Build Year')
```



This code uses the matplotlib library to create a histogram which show the house build on that city.

Question 2: Show the total number house sold on the different year and visualize it in bar graph by using seaborns library?

In [194…
```
# Count the total number of sales in different years and sort by year
salescount = df['Yr Sold'].value_counts().sort_index()
```

```
# Print or use the result as needed
print(salescount)
```

```
2006    601
2007    657
2008    593
2009    626
2010    341
Name: Yr Sold, dtype: int64
```

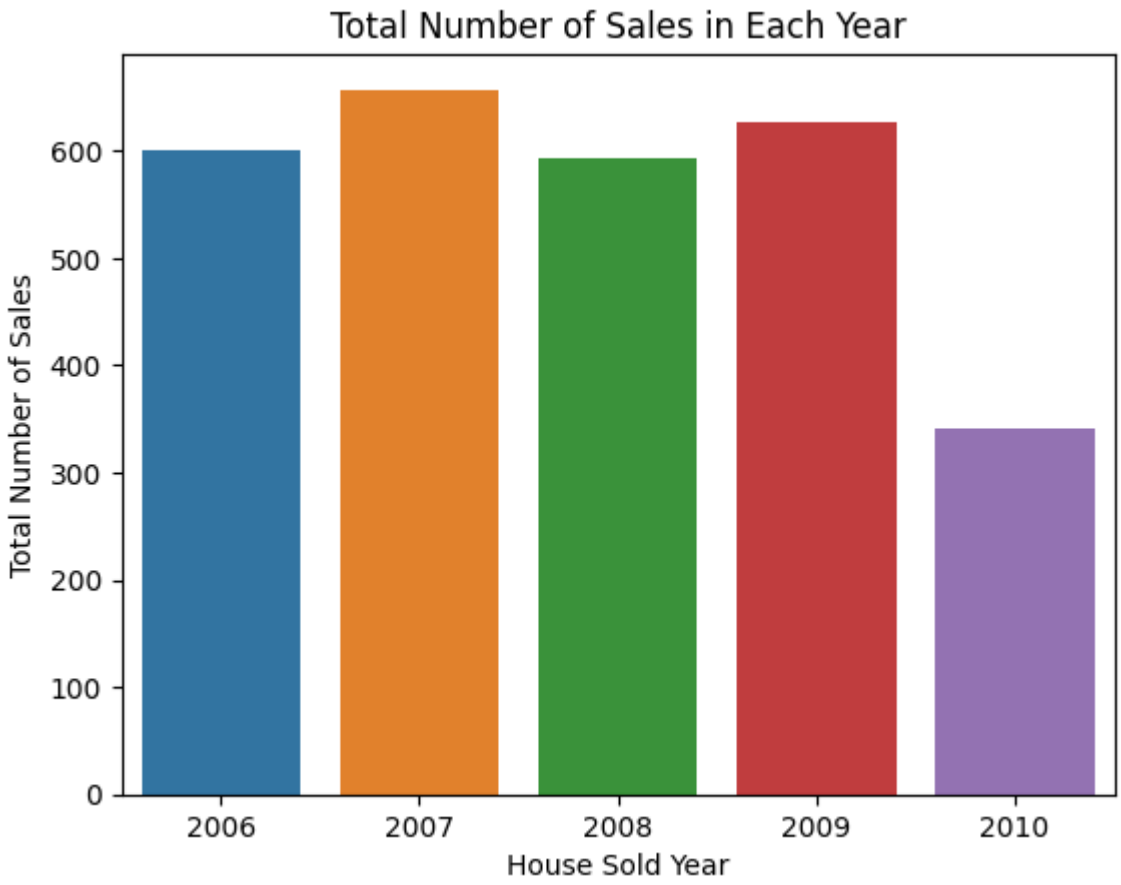
This code counts the total number of house sales in different year and store the information in the salescount dataframe. Now with the help of seaborns library these detials are visualize in bar graph.

In [195…

```
#Creating a bar graph with using seaborns, where x-asis as year that is index value
sns.barplot(x=salescount.index, y=salescount.values)

#Labeling the title, x-axis and y-axis
plt.title('Total Number of Sales in Each Year')
plt.xlabel('House Sold Year')
plt.ylabel('Total Number of Sales')

plt.show()
```



From this bar graph it is clear that on 2007 there was a higher number of sales whereas in 2010 least.

Question 3: Find out which state of dwelling property sales most? Visualize in line graph by using seaborn library.

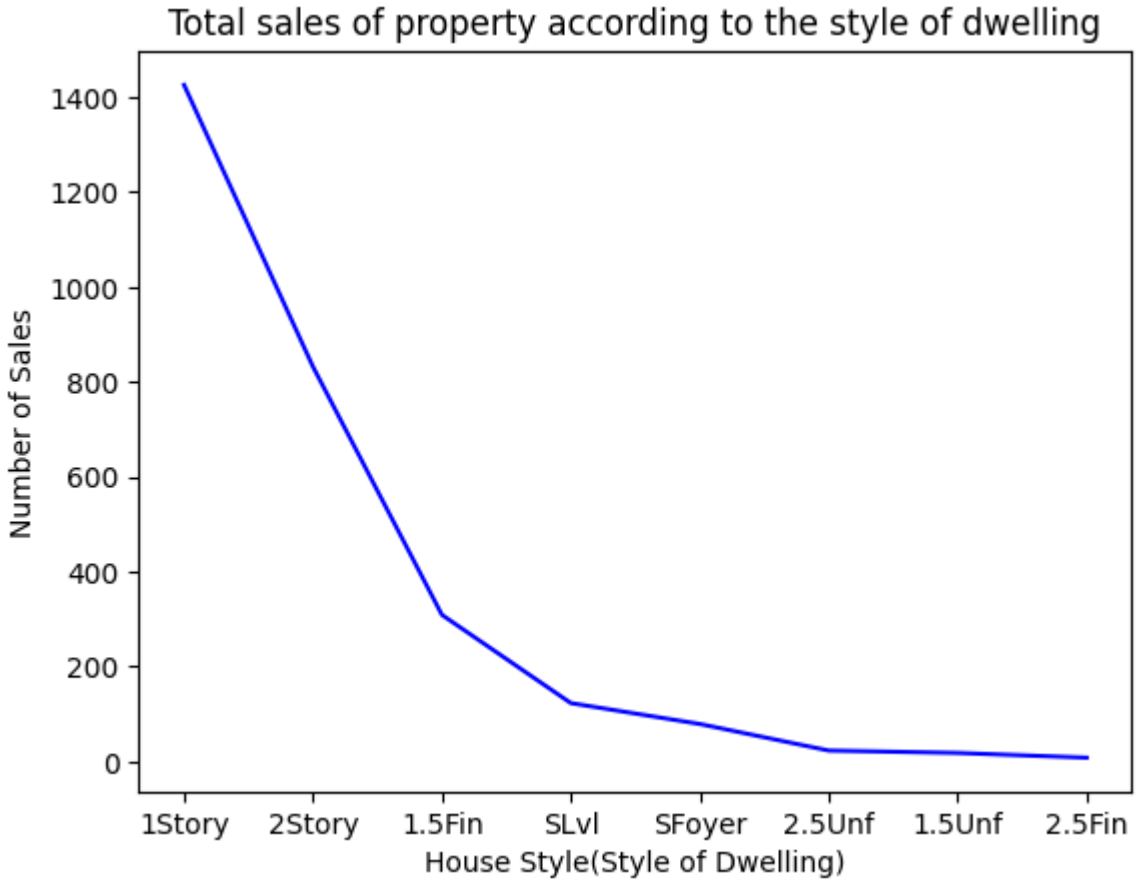
In [196…

```
#Count the total number of sales according to the dwelling property.
state_sales = df['House Style'].value_counts()
```

```
state_sales
```

Out[196]:
```
1Story    1426
2Story     832
1.5Fin     309
SLvl       123
SFoyer      79
2.5Unf      23
1.5Unf      18
2.5Fin       8
Name: House Style, dtype: int64
```

In [197…
```
sns.lineplot(x=state_sales.index, y=state_sales.values, color='blue')
plt.title('Total sales of property according to the style of dwelling')
plt.xlabel('House Style(Style of Dwelling)')
plt.ylabel('Number of Sales')
plt.show()
```



The result show that the one story dwelling sales the most whereas two and one-half story: 2nd level finished sales the least one.

Bivariate Analysis

Question 1: Show the relationship between sales price and GR LIV Area (Living area above ground in square feet). Visualize this data with the help of seaborn in scatter plot.

In [198…
```
#creating the scatter plot with two variables
sns.scatterplot(x='Gr Liv Area',
                y='SalePrice',
                data=df)
```

```
#Labeling title, x-axis and y-aixs
plt.title('Relationship Between Sales Price and Total Living Area')
plt.xlabel('Total Living Area (In Square feet)')
plt.ylabel('Sales Price (In Dollar)')
plt.show()
```

This result show that there is a direct relationship between the total area vs sales price that is the more total living area the property values the most while selling.

Question 2 : Show the relationship between sales price and GR LIV Area (Living area above ground in square feet) along with the property sold out year. Visualize this data with the help of seaborn in scatter plot

In [199…

```
#creating the scatter plot with three variables and year sold with different color
sns.scatterplot(
                x='Gr Liv Area',
                y='SalePrice',
                hue='Yr Sold',
                data=df,
                palette='viridis'
               )
#Labeling the title, x-axis, y-axis and year sold out
plt.title('Relationship Between Sales Price and Total Living Area With Property Sol
plt.xlabel('Total Living Area (In Square feet)')
plt.ylabel('Sales Price (In Dollar)')
plt.legend(title ='Year Sold')
plt.show()
```

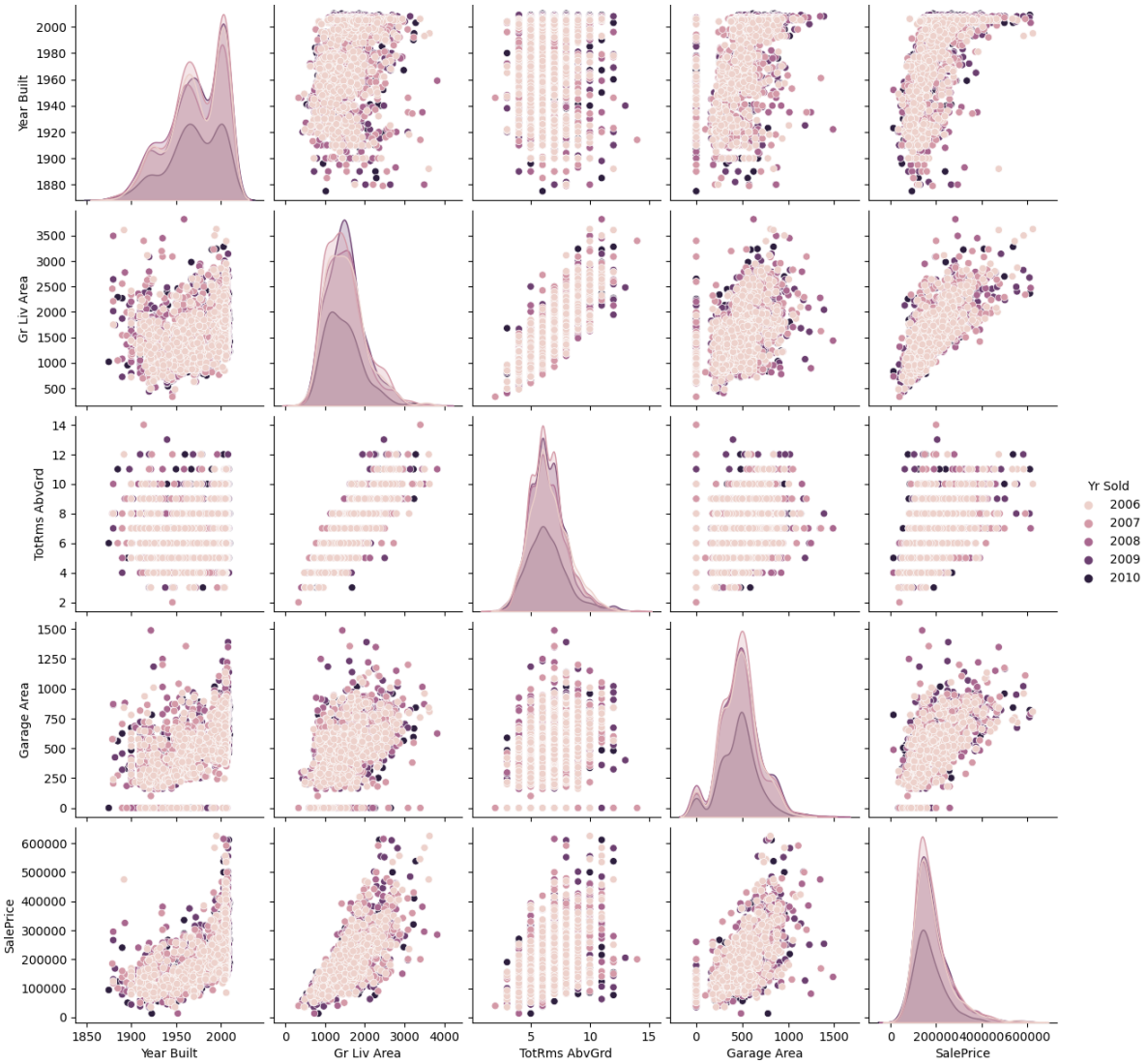
This graph shows that there is huge number of sales in 2006 either it has a lower price or higher price, or lower living area or higher living area.

Question 3 : Show the multiple relationship between various variables that is Year Built, Living Area, Total rooms, Garage Area, Sales Price and Year Sold in single pair plot?

In [200…

```
#creating a pair plot with different variables
sns.pairplot(data = df,
             vars = ['Year Built','Gr Liv Area', 'TotRms AbvGrd','Garage Area','Sal
             hue ='Yr Sold')

plt.show()
```



Question 4 : Show the correlation between various variables that is Year Built, Living Area, Total rooms, Garage Area, Sales Price and Year Sold. And also visaulize the data with heat map.

In [201…
```
#Calculate the correlation between different variables
result = df[['Year Built','Gr Liv Area', 'TotRms AbvGrd','Garage Area','SalePrice',
result
```

Out[201]:

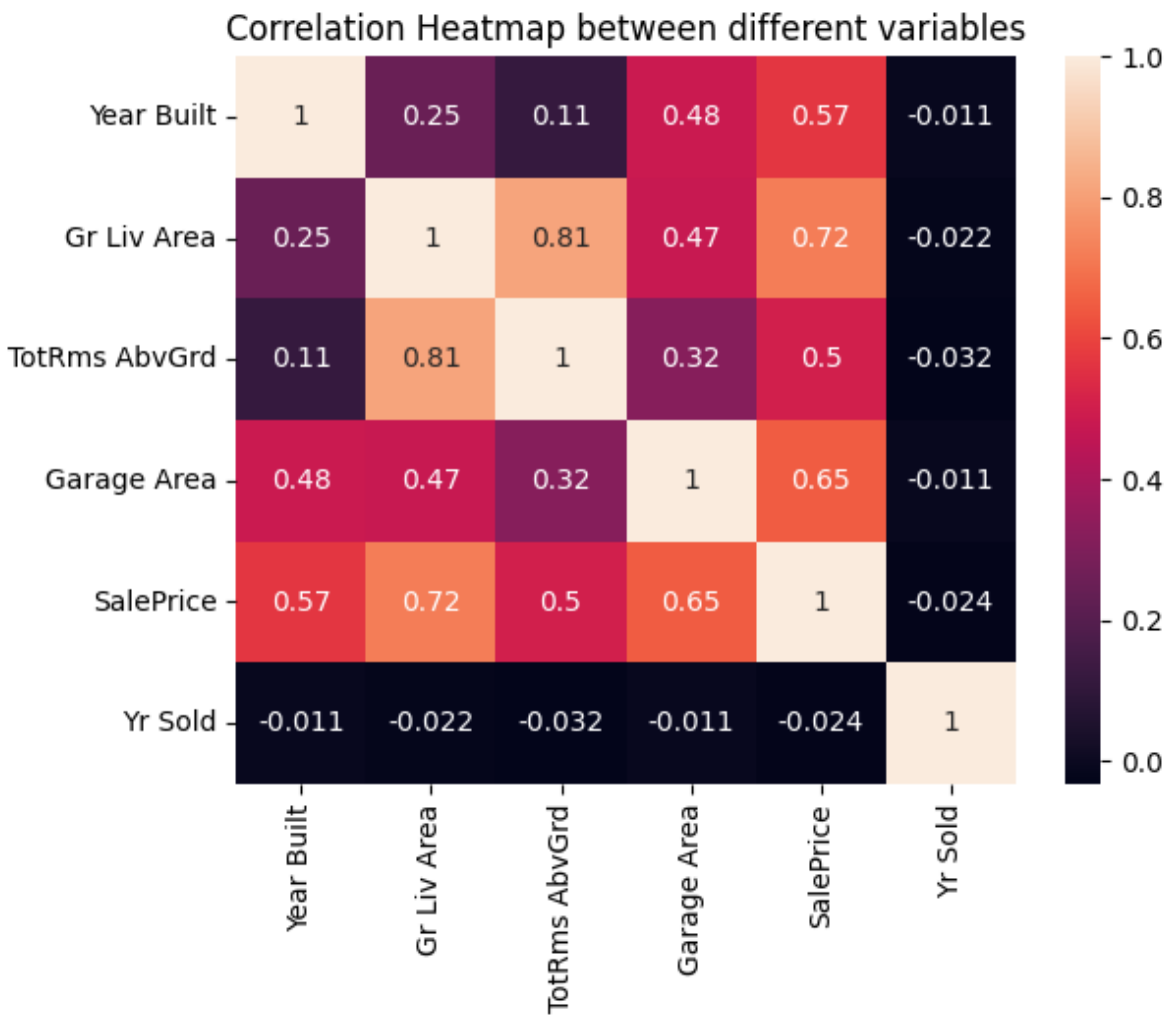
	Year Built	Gr Liv Area	TotRms AbvGrd	Garage Area	SalePrice	Yr Sold
Year Built	1.000000	0.249752	0.114330	0.482606	0.568136	-0.011355
Gr Liv Area	0.249752	1.000000	0.812161	0.474095	0.720965	-0.022311
TotRms AbvGrd	0.114330	0.812161	1.000000	0.317097	0.503028	-0.032277
Garage Area	0.482606	0.474095	0.317097	1.000000	0.647025	-0.011044
SalePrice	0.568136	0.720965	0.503028	0.647025	1.000000	-0.024288
Yr Sold	-0.011355	-0.022311	-0.032277	-0.011044	-0.024288	1.000000

After calculation of correlation let plot the data using heat maps.

In [202…
```
#Creating a heatmap with the help of correlation dataframe
sns.heatmap(result, annot=True)
```

```
#Labeling title
plt.title('Correlation Heatmap between different variables')

plt.show()
```



RESULTS AND DISCUSSION

Key Findings

There are alot of key finding through this analysis. Here are the few points:-

1. **Correlation Between Different Variables**: In this analysis correlation between Year Built, Living Area, Total rooms, Garage Area, Sales Price and Year Sold were calculated and visulaize.
2. **Missing Data**: There are alots of missing data which are handled with appropirate strategies.
3. **Handling Outliers and duplicate data**: Different outliers were handled with proper investigation and all the duplicate data were also removed.
4. **Data according to the time trends**: Build and sales of buildling were fiqure out with different visualization.
5. **Relationship between different varialbles**: Different graph was used to show the different relationship with each variables.

Limitations / Recommendations

The main limitation on dataset is missing values. There are lots of variables on which value was missing. So to overcome this issue, they should focused on the better data collection and also there are some outlier like as month values is more than 12, so while putting the details on the database proper information should put.

References

Pandas, 2023. pandas documentation. [Online] Available at: https://pandas.pydata.org/docs/ [Accessed 28 11 2023].

Python, n.d. Python 3.12.0 documentation. [Online] Available at: https://docs.python.org/3/ [Accessed 11 25 2023].

Seaborn, n.d. seaborn: statistical data visualization. [Online] Available at: https://seaborn.pydata.org/ [Accessed 24 11 2023].

Shaoyinger, 2017. Data Exploration and Prediction of House Price. [Online] Available at: https://www.kaggle.com/code/shaoyingzhang/data-exploration-and-prediction-of-house-price [Accessed 26 11 2023].